JOURNAL OF COMPUTER AND SYSTEM SCIENCES 31, 169–181 (1985)

Relativized Circuit Complexity

CHRISTOPHER B. WILSON

Department of Computer and Information Science, University of Oregon, Eugene, Oregon 97403

Received April 17, 1984; June 24, 1985

We compare the measures of sequential time modelled using Turing machines and of parallel size modelled using Boolean circuits. This is done by constructing oracles which show certain relationships between complexity classes. An oracle B is shown for which $\Delta_2^{P,B}$ has $2n + o(n)$ size circuits relative to B. On the other hand, we give a C so that P^C does not, for any k, have size n^k circuits relative to C and yet $NP^C \neq coNP^C$. These techniques can be combined to yield a D relative to which P^D has $2n + o(n)$ size circuits but R^D does not have size n^k circuits for any k. © 1985 Academic Press, Inc.

1. INTRODUCTION

1.1. *A Circuit-Based Oracle Model*

This paper examines the relationship of two complexity measures: Turing machine *time*, a uniform measure, and boolean circuit *size*, a nonuniform one. A major goal of complexity theory is to show either the separation or the equality of various classes. The former is typically done by using a *diagonalization* method and the latter through some form of *simulation.* Oracles have been used to show the limitations of these techniques, especially since [2]. However, *oracle techniques have been tailored to Turing machines*: to talk about relativized circuits, the circuits had to be simulated on a Turing machine based model. Consequently, a good deal of power has been lost due to that simulation. To examine the fine structure of relativized circuit complexity, an *intrinsically circuit-based oracle model* is required. The new model is introduced in this paper. The power of this approach is shown below.

1.2. *Review and Outline*

In examining POLY-SIZE (the class of sets accepted by polynomial size families of circuits) versus the polynomial hierarchy, it turns out that we know very few concrete facts. Certainly we know that $P \subseteq$ POLY-SIZE, and Adleman [1] has shown that $R \subseteq$ POLY-SIZE (later improved to BPP in [4]). In terms of lower bounds, Blum [5], following [15], has exhibited a set in P which requires circuit size $3n$. This is so far the best known, so it is still open as to whether any sets in P have non-linear lower bounds on circuit size. Kannan [11] has shown that for all k, $\sum_2^P \cap \prod_2^P$ is not contained in SIZE (n^k), although this does not rule it out of

0022-0000/85 $3.00

Copyright © 1985 by Academic Press, Inc.
All rights of reproduction in any form reserved.

being in POLY-SIZE. Also, if f is a super-polynomial function, then $\sum_2^f \cap \prod_2^f$ is not in POLY-SIZE.

Karp and Lipton [12] present various interesting results, one of the nicer ones being that if NP has polynomial size circuits, then the polynomial hierarchy collapses to $\sum_2^P \cap \prod_2^P$. Thus, our current situation is as follows:

FACT I. BPP $\subseteq$ POLY-SIZE.

FACT II. (i) $\forall k$, $\sum_2^P \cap \prod_2^P \not\subseteq$ SIZE (n^k)

(ii) f *super-polynomial* $\Rightarrow \sum_2^f \cap \prod_2^f \not\subseteq$ POLY-SIZE.

FACT III. $NP \subseteq$ POLY-SIZE $\Rightarrow PH = \sum_2^P \cap \prod_2^P$.

These results seem hard to improve. Indeed, a consequence of this paper is that they will not be improved by means of a relativizable proof technique.

In Section 2, we introduce a model of relativized circuits. Using this model, we are able to obtain the following results:

(a) There exists an oracle B such that $\Delta_2^{P,B}$ has bounded linear size circuits relative to B.

This tells us that we will not be able to obtain non-linear lower bounds for the circuit size of sets in P (or NP or Δ_2^P for that matter) with a proof technique which relativizes. The oracle provides an interesting structural contrast of $\Delta_2^{P,B}$ with $\sum_2^{P,B} \cap \prod_2^{P,B}$, which does not have size n^k for circuits for any fixed k. And because of that, the result of Karp and Lipton cannot be improved by a relativizable proof.

(b) There exists an oracle C such that for no fixed integer k does P^C have n^k size circuits relative to C (and yet $NP^C \neq NP^C$).

Due to the difficulty of obtaining non-linear lower bounds for sets in P, one might be temped to prove the opposite. The oracle C indicates that this too would require a non-relativizable proof technique. It is easy to show that if $P = NP$, then P would not have size n^k circuits for any fixed k. The fact that $NP^C \neq coNP^C$ shows what can happen in a case where the polynomial hierarchy does not collapse.

(c) There exists an oracle D such that P^D has bounded linear circuit size but R^D does not have n^k size circuits relative to D for any fixed k.

This shows a strong relativizable structural difference between the classes P and R (random polynomial time). Both classes we already know to have polynomial size circuits. Here, the expected extra power of R is witnessed by the polynomial gap in their circuit sizes.

2. DEFINITIONS AND THE NEW MODEL

We say that $t(n)$, t: $N \to N$, is a *time bound* if t is a strictly increasing function such that $t(n) \geq n$ almost everywhere. Furthermore, we require that $t(n)$ be a constructible function. A machine has time bound $t(n)$ if the number of steps executed

by the machine on any input of length n is no greater than $t(n)$ for sufficiently large n. A function g is said to be $O(f)$ if g is bounded above by a constant multiple of f almost everywhere. A function g is $o(f)$ if the limit as n approaches ∞ of $g(n)/f(n)$ is 0.

In order to relativize the classes of interest, we have to settle on what kind of computational query device is to be used and appropriate ways to measure its complexity. Now the device used to examine relativized versions of classes such as P and NP has long since been decided: the *oracle Turing machine*. Its complexity is simply the running time with a query taking one step. The same model is used to relativize PSPACE, although there is no agreed on definition of the proper space measure, the issue being whether to or how to count the use of the oracle tape.

For a discussion of oracle Turing machines and some of the original relativizations of the polynomial hierarchy, here abbreviated to PH, the reader is referred to [2] for an excellent exposition of this subject. Recall that, for an oracle X, $\Delta_2^{P,X}$ is defined as the union over $Y \in NP^X$ of P^Y. So $NP^X \subseteq \Delta_2^{P,X} \subseteq \sum_2^{P,X} \cap \prod_2^{P,X}$.

To deal with more general hierarchies such as the exponential time analog of the polynomial hierarchy, we use the idea of a $\sum_k$ machine. Such a machine can make existential and universal moves; note that a nondeterministic machine can make only existential moves. It must start out in an existential mode, making only existential moves, and can alternate modes at most $k-1$ times. A complementary notion of a $\prod_k$ machine would start in a universal mode. A set is in $\sum_k^{T(n)}$ if it is accepted by a $\sum_k$ machine all of whose computation paths have length at most $T(n)$ on an input of length n. In this manner we define classes such as $\sum_2^{\mathrm{EXP}}$, where EXP denotes functions of the form $2^{n^{o(1)}}$.

The notion of a probabilistic Turing machine and random polynomial time has been introduced in [7]. A language L is in R if there is a polynomial time probalilistic Turing machine—one that can flip an unbiased coin—such that if $x \in L$, then M accepts x with probability at least $\frac{1}{2}$; if $x \notin L$, then M rejects x. Notice that only one-sided error is allowed. A related class, BPP, allows two-sided error. So $L \in \mathrm{BPP}$ if there is a fixed $\varepsilon > 0$ and a machine M as above such that M gives the correct answer with probability greater than $\frac{1}{2} + \varepsilon$. For both R and BPP, the probability of error can be made arbitrarily small by multiple runs of the machine. It is clear from the definitions that $P \subseteq R \subseteq \mathrm{BPP}$. Another characterization of an $L \in R$ in terms of nondeterministic Turing machines is that L can be accepted by such a machine, but if there is an accepting computation path, then some fixed fraction of the computation paths must be accepting. So it is easily seen that $R \subseteq NP$. BPP is somewhat trickier, but it has been shown that $\mathrm{BPP} \subseteq \sum_2^P \cap \prod_2^P$ [18, 13].

Here, to relativize circuit size, we have had to introduce the notion of *relativized* circuits. A relativized circuit will have n inputs (in terms of which its size will be measured), generally one output, and is built up from binary logical gates (and, or, not, 1, 0) and oracle gates. An oracle gate is a k-input 1-output gate which on input x of length k outputs 1 if x is in the given oracle and 0 otherwise.

DEFINITION. SIZE $^A(t(n))$ will be the set of those languages L for which there is

a family of circuits $\{\alpha_n\}$ relative to the oracle A such that $L \cap \{0, 1\}^n$ is the set of strings accepted by α_n and each α_n has no more than $t(n)$ edges.

DEFINITION. For any oracle A, POLY-SIZE^A is $\bigcup_k$ SIZE$^A(n^k)$.

This model has been introduced and has had some properties analyzed in [19]. For example, the existence of an oracle X was shown such that NP^X is contained in SIZE$^X(2n + o(n))$. Note that for this oracle, $NP^X \neq coNP^X$, because NP^X must differ from $\sum_2^{P,X} \cap \prod_2^{P,X}$ by Kannan's result, Fact II(i). It is worth noting that prior to this, Rackoff, see [17], constructed an oracle D such that $P^D \neq R^D = NP^D$. So NP^D has polynomial size circuits relative to D and yet $P^D \neq NP^D$. Immerman and Mahaney in [10] were able to exhibit an A so that NP^A has polynomial size circuits relative to A while $NP^A \neq coNP^A$.

Also, this model was shown to satisfy a relativizable analog of Lupanov's circuit size lower bound as follows:

THEOREM 0. *Consider the functions f: $\{0, 1\}^n \to \{0, 1\}$. Let A be any oracle. If questions to A of size no greater than $p \geq 2$ (perhaps a function of n) are allowed, then there is a function f such that the size of any circuit family relative to A computing it must be at least*

$$\frac{2^n}{(p-1)\,n} - o\left(\frac{2^n}{(p-1)\,n}\right).$$

3. MAIN THEOREMS

The central role of Δ_2^P and $\sum_2^P \cap \prod_2^P$ in the comparison of uniform and nonuniform complexity measures is emphasized by the following facts:

—If there is a sparse set $\leq_T^P$-hard for NP (that is, $NP \subseteq$ POLY-SIZE), then $PH = \sum_2^P \cap \prod_2^P$ [12].

—If there is a sparse set $\leq_T^P$-complete for NP, then $PH = \Delta_2^P$ [14].

We note that if $\sum_2^P \neq \prod_2^P$, then NP does not have polynomial size circuits, for otherwise the hierarchy would collapse. Baker and Selman in [3] have shown the existence of an oracle under which this is the case. So we have the following.

PROPOSITION 1. *There is an oracle A such that NP^A does not have polynomial size circuits relative to A.*

Interestingly, Heller [8] has shown that $\sum_2^P \neq \prod_2^P$ relative to most (i.e., all but an effectively nowhere dense subset of the) *recursive* oracles. An open question is whether $\sum_2^P \neq \prod_2^P$ with probability 1 (here meaning relative to a random oracle) as in [4], which seems likely. This could be taken as evidence that $NP \not\subseteq$ POLY-SIZE.

Certainly, Proposition 1 tells us that the result BPP $\subseteq$ POLY-SIZE is not going to be improved by a relativizable proof technique showing that some higher levels of the polynomial hierarchy, such as $\sum_2^P$, are contained in POLY-SIZE.

An interesting issue is whether one can establish non-linear lower bounds on the circuit size of sets in P or NP. That is, we would like to improve Fact II, which states that for all k, $\sum_2^P \cap \prod_2^P \not\subseteq \text{SIZE}(n^k)$, by replacing $\sum_2^P \cap \prod_2^P$ with Δ_2^P. Theorem 3.1 will show that any such improvement will require a proof that will not relativize. First, we present a technical lemma.

LEMMA 2. *Given T, a time bound, there exists an oracle B such that for all $c, k \geqslant 1$,*

$$\Delta_2^{T(cn^k),B} \subseteq \text{SIZE}^B(\log T(cn^k + o(n^k)) + cn^k + n + o(n^k)).$$

The statement of the lemma is complicated, but it has as immediate corollaries some pleasant theorems which follow.

Proof of Lemma 2. The oracle is shown to exist by construction. Let M_i be an enumeration of the deterministic query machines with time bounded by $T(c_i n^{k_i})$, and let NM_i be an enumeration of nondeterministic machines.

We characterize $\Delta_2^{T(\text{poly})}$, where $T(\text{poly})$ is the set of functions of the form $T(cn^k)$, by defining a complete set as in [9]:

$$K(B) = \{\langle j, y, 0^m\rangle : NM_j^B \text{ accepts } y \text{ within } m \text{ steps}\}.$$

To say that $L \in \Delta_2^{T(\text{poly})}$ is to say that there is a machine M_i which, with the oracle $K(B)$, accepts L in time bounded by a function in $T(\text{poly})$.

To arrange that the class $\Delta_2^{T,B}$ has small circuits, we will examine all $\langle i, x\rangle$ (for a suitable integer encoding $\langle \cdot, \cdot \rangle$) of a particular length, find a short α, and put $\langle i, x\rangle\alpha$ into either B or $\bar{B}$. We will construct the oracle so that

$$\forall n\, \exists \alpha\, \forall i\, \forall x,\ |\langle i, x\rangle| = n,\ M_i^{K(B)} \text{ accepts } x \Leftrightarrow \langle i, x\rangle\alpha \in B.$$

The query $\langle i, x\rangle\alpha$ will form the small circuit accepting $L \cap \{0, 1\}^n$. Thus, i and α will be hardwired into the circuit and x will be the input to the circuit. Thus, the circuit looks like Fig. 1. We must ensure that such an α exists and that it grows sufficiently slowly as a function of $|x|$.

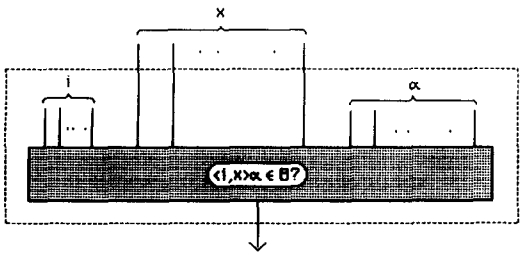

FIGURE 1

CONSTRUCTION OF B. Stage 0. Initially, B is empty.
Stage N. *Step* 1. Examine each $\langle i, x\rangle$, $|\langle i, x\rangle| = n$, where $c_i n^{k_i} = N$.

(a_0) Run M_i on x for $T(N)$ steps, handling oracle queries as described in step (a_1).

(a_1) When any $\langle j, y, 0^m\rangle$ is queried, determine if $\langle j, y, 0^m\rangle \in K(B)$ (by running NM_j^B on y for m steps). If $\langle j, y, 0^m\rangle \notin K(B)$, we will see if we can force NM_j^B to accept y by adding certain strings to B. Strings we may add to B must be previously unreserved and of length at most $m < T(N)$. If there are at most m of these strings which, when added to B, will cause NM_j^B to accept y within m steps, then we place these strings in B. Now if at this point $\langle j, y, 0^m\rangle \in K(B)$, we choose some accepting computation of NM_j^B on y and reserve for $\bar{B}$ the at most m unreserved strings queried on that computation. Returning to the simulation in step (a_0), answer the query appropriately.

(b) If $M_i^{K(B)}$ now accepts x, put $\langle i, x\rangle$ into S_a. Otherwise, put $\langle i, x\rangle$ into S_r (and note that further changes to B cannot cause any NM_j^B to change its behavior on y, so the strings in $K(B)$ queried by M_i are immune to change). The sets S_a and S_r will be coded into B.

Step 2. Choose an α of length $\log T(N) + N + 3$ such that no member of $S_a\alpha$ has been reserved for $\bar{B}$ and no member of $S_r\alpha$ for B. Put all of $S_a\alpha$ into B, reserve all of $S_r\alpha$ for $\bar{B}$, and reset S_a and S_r to the empty set.

To complete the proof, we note in Step 2, when each $\langle i, x\rangle \in S_a$ is encoded into B, that $M_i^{K(B)}$ accepts x within its time bound. Furthermore, no change to the oracle that could be made in later stages can alter the behavior of $M_i^{K(B)}$ on x as we have fixed the queries made to $K(B)$. Also, any $\langle i, x\rangle$ will be in exactly one of S_a or S_r.

Consider any $L \in \Delta_2^{T(\text{poly}),B}$. L is accepted by some $M_i^{K(B)}$ for some fixed integer i. We must show that for each n, $L^n = L \cap \{0, 1\}^n$ is accepted by a single circuit. For all x, $|x| = n$, the encoding $\langle i, x\rangle$ will have constant length, $|\langle i, x\rangle| = m$. Let α_N be the string chosen at stage N of the construction, where $N = c_i m^{k_i}$. By the discussion above,

$$\begin{aligned} x \in L &\Leftrightarrow M_i^{K(B)} \text{ accepts } x \text{ within its time bound} \\ &\Leftrightarrow \langle i, x\rangle \in S_a \text{ at stage } N \\ &\Leftrightarrow \langle i, x\rangle\alpha_N \in B. \end{aligned}$$

The single circuit accepting L^n is described by $\lambda x.\ \langle i, x\rangle\alpha_N$, as pictured above.

It remains to show that the α chosen in Step 2 exists and that the size of the resulting circuit is within the desired size bound.

Each of the at most 2^{N+1} $(> \sum_{i=1}^{N} 2^i)$ strings $\langle i, x\rangle$ will cause at most $T(N)$ strings to be reserved for B or $\bar{B}$ in Step 1 and one string in Step 2. Thus, at the

beginning of Step 2 of some stage N, the total number of reserved strings is less than

$$\begin{aligned}&\left\lceil \sum_{k=1}^{N-1} 2^{k+1}(T(k)+1) \right\rceil + 2^{N+1} T(N)\\ &\quad < (T(N-1)+1)\, 2^{N+1} + 2^{N+1} T(N)\\ &\quad < 2^{N+2} T(N) + 2^{N+1}\\ &\quad < 2^{N+3} T(N) = 2^{\log T(N) + N + 3}.\end{aligned}$$

So there will always be an α satisfying the requirements of Step 2.

The size of the circuit $\langle i, x\rangle\alpha$ is

$$n + |\alpha| = \log T(c_i n^{k_i}) + c_i n^{k_i} + n + 3.$$

But actually we must examine $n = |\langle i, x\rangle|$ in terms of $|x|$. The coding can be done efficiently, so we will say that $|\langle i, x\rangle| = |x| + o(|x|)$. So the size of $\langle i, x\rangle\alpha$, in terms of $n = |x|$, is $\log T(c_i n^{k_i} + o(n^{k_i})) + c_i n^{k_i} + n + o(n^{k_i})$. Q.E.D.

THEOREM 3.1. $\exists B,\ \Delta_2^{P,B} \subseteq \mathrm{SIZE}^B(2n + o(n))$.

THEOREM 3.2. $\exists B,\ \Delta_2^{\mathrm{EXT},B} \subseteq \bigcup_c \mathrm{SIZE}^B(cn)$, *where EXT denotes functions of the form* $2^{O(n)}$.

THEOREM 3.3. $\exists B,\ \Delta_2^{\mathrm{EXP},B} \subseteq \mathrm{POLY\text{-}SIZE}^B$, *where EXP denotes functions of the form* $2^{n^{o(1)}}$.

The first theorem follows immediately by setting $T(n) = n^{\log n}$ and $c = k = 1$. Notice the contrast with Kannan's result, Fact II(i) above. First we see that the fact cannot be improved by a relativizable proof technique to show that Δ_2^P is not in $\mathrm{SIZE}(n^k)$ for any k. Also note that this oracle B separates $\Delta_2^{P,B}$ from $\sum_2^{P,B} \cap \prod_2^{P,B}$ ($= PH^B$ by Fact III). Such an oracle has been constructed directly by Heller [8], but the interesting thing about this oracle is that it exhibits a *very* strong relativizable separation of Δ_2^P and $\sum_2^P \cap \prod_2^P$.

Recall Karp and Lipton's result, Fact III, that if NP has polynomial size circuits, them the PH collapses to $\sum_2^P \cap \prod_2^P$. Theorem 3.1 also shows, due to the separation mentioned above, that even a minor improvement in the right hand side of Fact III is unlikely, and this would still be true if the left hand side was the far stronger assumption that Δ_2^P had (bounded) *linear* size circuits. Further, the theorem dashes any hopes of easily exhibiting non-linear circuit-size lower bounds for sets low in the polynomial hierarchy.

The theorem seems somewhat surprising in view of Blum's lower bound of a set in P requiring circuit size $3n$. This points out an example of a proof technique which does not relativize. Another such nonrelativizable result is found in [16], where it is shown that linear nondeterministic time properly contains linear deter-

ministic time. Both of these were based on difficult combinatorial arguments, which do not seem to apply well to the relativized domain.

The last two theorems show what can happen to hierarchies more complex than polynomial time. Theorem 3.2 is interesting in view of fact II(ii). Here we have an oracle B for which $\Delta_2^{\mathrm{EXT},B}$ has linear size circuits but $\sum_2^{\mathrm{EXT},B} \cap \prod_2^{\mathrm{EXT},B}$ does not even have polynomial size circuits. Also, note that $\sum_2^{P,B} \cap \prod_2^{P,B}$ cannot be contained in $\Delta_2^{\mathrm{EXT},B}$.

Theorem 3.3 exhibits the most complex set that we are able to give polynomial size circuits under relativization. For this oracle, PSPACE^B (here the query tape is counted in the space bound) also will be in $\mathrm{POLY\text{-}SIZE}^B$, and yet $NP^B \neq coNP^B$ (because $\Delta_2^{\mathrm{EXP},B} \neq \sum_2^{\mathrm{EXP},B} \cap \prod_2^{\mathrm{EXP},B}$).

As is usually the case with this approach, we can relativize the opposite state of affairs. That is, there will be no relativizable proof that Fact II(i) is the best possible, which we show by exhibiting an oracle for which P does have non-linear lower bounds on its circuit size. A simple way to do this is to pick an oracle relative to which $P = NP$, and then $P = \sum_2^P \cap \prod_2^P$ so for all k, $P \not\subseteq \mathrm{SIZE}(n^k)$. But this is the trivial case, and we want to know how complex (nonuniformly) sets in P can be if $P \neq NP$.

THEOREM 4. *There exists an oracle C such that*

$$NP^C \neq coNP^C$$

and

$$\forall k,\ P^C \not\subseteq \mathrm{SIZE}^C(n^k).$$

The proof of Theorem 4 makes use of the following

FACT. *For sufficiently large n, the number of subsets of $\{0,1\}^n$ accepted by circuits of size at most n^k relative to a fixed oracle is bounded by $2^{n^{2k+1}}$.*

Proof of Fact. In [19], we saw that the number of functions computed by size N circuits with the length of the oracle query bounded by p is at most m^m, where $m = N(p-1) + o(N)$. Letting $N = n^k$ and $p = n^k$, we see that

$$m^m < (n^{2k})^{n^{2k}} < 2^{n^{2k+1}}$$

as n gets sufficiently large. Interestingly, this is the same as a result in [11]. Q.E.D.

Proof of Theorem 4. Define the diagonal sets as follows:

$$L(C) = \{x: \exists y,\ |y| = |x|,\ xy \in C\} \text{ and}$$
$$L^k(C) = \{x: x0^{|x|^{2k+2}} \in C\}.$$

Clearly, for all oracles C, $L(C) \in NP^C$ and $L^k(C) \in P^C$. As we construct our C, we ensure that $\overline{L(C)} \notin NP^C$ and that no family of n^k-sized circuits will accept $L^k(C)$.

Also, let NM_i be an enumeration of the nondeterministic oracle machines with polynomial run-time bounds $p_i(n)$.

The construction varies between two steps, the first ensuring that each NM_i^C does not accept $\overline{L(C)}$, the second that no circuit of size $\leqslant n^k$ accepts $L^k(C)$.

CONSTRUCTION OF C. Initially $C \leftarrow 0$, $n \leftarrow 1$, and $k \leftarrow 0$.

For $i = 0$ to ∞ do:
Step 1:

Increase n so that
n is larger than the length of any string queried or reserved at any earlier step and $p_i(n) < 2^n$
Run NM_i^C on 0^n and if it accepts,
then choose an accepting path and some y ($|y| = n$) where $0^n y$ was not queried on that path, and put $0^n y$ into C
else do nothing ($0^n \notin L(C)$)
note: NM_i^C accepts $0^n \Leftrightarrow 0^n \notin \overline{L(C)}$

Step 2: $k \leftarrow k + 1$

Increase n so that
n is larger than the length of any string queried or reserved at any earlier step and $2^n > n^{2k+1}$
Until all n-input 1-output circuits of size $\leqslant n^k$ are cancelled
choose an unused x of length n
run all uncancelled circuits on x
at least half either accept or reject, choose the majority
if it is a rejecting majority, put $x0^{n^{2k+2}}$ into C
mark x as used and cancel the appropriate majority
end / *until loop* /
end / *for loop* /

There are $2^{n^{2k+1}}$ sets accepted by circuits of size $\leqslant n^k$, so we need to use n^{2k+1} strings x of length n to cancel all circuits. Since the number of possible strings x, 2^n, is larger than the number we need, we will always be able to find an unused one in the loop of Step 2.

No circuit of size $\leqslant n^k$ can query a string of length greater than n^k, so adding $x0^{n^{2k+2}}$ to C in Step 2 will not affect the behavior of the circuits. Q.E.D.

In proving that $NP^C \neq coNP^C$, we only wanted the hierarchy not to collapse to P, something which could be accomplished a number of ways. Notice that due to the spreading of the diagonalization, this theorem could easily have been restated. We could have shown, instead of $NP^C \neq coNP^C$, that $P^C \neq NP^C$. Similarly, we could have strengthened that condition to $\sum_2^{P,C} \neq \prod_2^{P,C}$ using the method of [3].

That would have given us an oracle C relative to which P would not have size n^k circuits for any k, yet NP would not be in POLY-SIZE.

It turns out that the situation of Theorem 4 holds with respect to a random oracle. [4] have shown that $NP^C \neq coNP^C$, for random C, with probability one. Gasarch [6] was able to show that these $L^k(C)$ are not contained in $\text{SIZE}^C(n^k)$ with probability one. This is due to the fact that since a circuit of size n^k cannot query a string x of length n^{2k+2}, the question of whether the circuit accepts such an x is independent of its membership in $L^k(C)$.

As mentioned earlier, we were able to show a relativizable separation of Δ_2^P and $\sum_2^P \cap \prod_2^P$ in a fairly strong sense. Another pair of classes can be similarly separated by interlacing the proofs of Lemma 2 and Theorem 4.

THEOREM 5. *There exists an oracle D such that*

$$P^D \subseteq \text{SIZE}^D(2n + o(n))$$

and

$$\forall k,\ R^D \not\subseteq \text{SIZE}^D(n^k).$$

Proof of Theorem 5. The diagonal set is here defined to be

$$T^k(D) = \{x: \exists y,\ |y| = |x|^{2k+2},\ xy01^k \in D\}.$$

Furthermore, each $T^k(D)$ will be guaranteed to have the property that $x \in T^k(D) \Leftrightarrow$ at least half the y's, $|y| = |x|^{2k+2}$, satisfy $xy01^k \in D$. Hence, $\forall k$, $T^k(D) \in R^D$. The 01^k signature will be used to ensure unique parsing of the string.

The construction of the oracle proceeds in stages. At each stage we decide the fates of strings up to a particular length. Each stage consists of two steps: in the first step we ensure that P^D will have small circuits, in the second step we diagonalize across the small size circuits to ensure that no n^k-sized circuit family can accept $T^k(D)$. To avoid conflict, Step 1 will add to D only strings of the form $z0$ and Step 2 strings of the form $z1$, though each step may reserve strings of any form for $\bar{D}$.

As before, we let M_i be an enumeration of the deterministic query machines, here with polynomial run-time bounds. Without loss of generality, we will assume that each machine will finish its computation on inputs of length n within $n^{\log n}$ steps.

The small circuits that we give to P^D are almost identical to those constructed in the proof of Theorem 3. After finding an appropriate α, we put strings of the form $\langle i, x\rangle \alpha 0$ into D to answer the query in the circuit (which will consist of a single query gate).

CONSTRUCTION OF D.
Stage 0: $D \leftarrow 0$ and $k \leftarrow 1$
Stage n:
    Step 1: $S \leftarrow 0$

Look at each $\langle i, x\rangle$ of length n
  run M_i^D on x for at most $n^{\log n}$ steps
  reserve for $\bar{D}$ all unreserved strings queried
  if M_i^D accepted x, then put $\langle i, x\rangle$ into S
  end

  Choose a string α of length $n + \log^2 n + 1$ such that no member of $S\alpha 0$ has been reserved for $\bar{D}$ and place all of $S\alpha 0$ into D

  Step 2: $m \leftarrow \lfloor n^{(2k+1)^{-1}} \rfloor$

If m is nice, that is, if the following **preconditions** hold:
  (1) m is larger than the length of any string reserved at any previous Step 2,
  (2) $m^{2k+2} > m^{2k+1} + (2k+1)^2 \log^2 m + 2 = n + \log^2 n + 2$, and
  (3) $2^m > m^{2k+1}$
then perform this step.

To perform this step:

Until all m-input circuits of size $\leqslant m^k$ are cancelled
  choose an unused x of length m
  run all uncancelled circuits on x
  determine the majority reaction (accept/reject)
  reserve for $\bar{D}$ all unreserved strings queried by the majority
  if it is a rejecting majority, put x into $T^k(D)$ later
  mark x used and cancel majority
  end / *until* /

To put each required x into $T^k(D)$,
  place all unreserved elements of $H(x) = \{xy01^k : |y| = m^{2k+2}\}$ into D
$k \leftarrow k + 1$

Any earlier Step 2 will not interfere with the determination of an α according to the requirements of Step 1. This is because in any Step 2 of a stage l, no string of length greater than l gets reserved for $\bar{D}$, and any string reserved for D ends in a 1.

Any Step 1 at a stage n will reserve at most $n^{\log n} 2^n = 2^{n + \log^2 n}$ strings. If all previous Step 1's are examined as well, they will have reserved less than $2^{n + \log^2 n + 1}$ strings. So in Step 1, choosing a satisfactory α of length $n + \log^2 n + 1$ will always be possible.

Precondition 1 of the niceness of m ensures that no previous Step 2 interferes with the construction in the current Step 2.

Precondition 3 guarantees that there will be enough (unused) strings x to complete the *until* loop of Step 2 (as in the proof of Theorem 4).

It remains to prove that, for each required x in Step 2, enough elements of $H(x)$ are unreserved. Now Step 1 has reserved less than $2^{n + \log^2 n + 1}$ strings, and the *until* loop of Step 2 has caused nothing longer than $m^k < m^{2k+1} = n$ to be reserved (so the latter strings can be ignored). We need more than half of $H(x)$ to be unreserved. Now $\text{card}\,(H(x))/2 = 2^{m^{2k+2}-1} > 2^{n + \log^2 n + 1}$ by Precondition 2. So the desired

property of $T^k(D)$ is maintained. Note that Step 1 cannot force an undesired string into $T^k(D)$, since strings placed into D then end in 0. Nor can another Step 2 (manipulating a $T^{k_0}(D)$ for $k_0 \neq k$) force undesired elements into $T^k(D)$ since strings placed into D then would lack the 01^k signature.

Clearly, $T^k(D) \notin \text{SIZE}^D(n^k)$ by the diagonalization done in the construction. As in the proof of Theorem 2, we have constructed small circuits (size $n + |\alpha| = 2n + \log^2 n + 1$) in term of the encoding $|\langle i, x \rangle|$. But since the encoding can be efficient, this translates into size $2n + o(n)$ circuits for languages in P^D.

Q.E.D.

4. Conclusions

By exhibiting relativizations of different relationships between members of the *PH*, we have shown that current (i.e., relativizable) proof methods will be insufficient to handle certain questions of uniform versus non-uniform complexity measures. Also, the existence of these oracles provides intuition as to what relationships are logically possible.

Many open questions and possibilities for further research remain:

(1) In what ways can $NP \cap coNP$ have (or not have) small circuits?

(2) Can we get a strong separation of *log-space* and P or of NP and Δ_2^P?

(3) Which relations hold under a random oracle?

Another possible extension would be to examine relativized circuit depth. This, however, seems to present the same messy problems as does relativized space for Turing machines. No measure seems particularly appropriate. Given an oracle node, would it have depth one? Or perhaps we would want to charge $\log s$, where s is the size of the query.

Acknowledgments

I would like to thank Charles Rackoff for many original ideas and countless hours of useful discussion. Silvio Micali also provided much help in the writing of this paper. The many helpful comments from the referees should likewise be noted. This research was carried out while the author was at the University of Toronto.

References

1. L. Adleman, Two theorems on random polynomial time, *in* "Proc. 19th IEEE Found. Comput. Sci., (1978), pp. 75–83.
2. T. Baker, J. Gill, and R. Solovay, Relativizations of the $P = ?NP$ question, *SIAM J. Comput.* **4**, No. 4 (1975), 431–452.

3. T. BAKER AND A. SELMAN, A second step toward the polynomial hierarchy, *in* "Proc. 17th IEEE Found. Comput. Sci., (1976)," pp. 71–75.
4. C. BENNETT AND J. GILL, Relative to a random oracle A, $P(A) \neq NP(A) \neq \text{co}NP(A)$ with probability 1, *SIAM J. Comput.* **10**, No. 1 (1981).
5. N. BLUM, "A Boolean Function Requiring $3n$ Network Size," Tech. Report A82/13, June 1982, Universität des Saarlandes, West Germany.
6. W. GASARCH, private communication.
7. J. GILL, Computational complexity of probabilistic Turing machines, *SIAM J. Comput.* **6**, No. 4 (1977), 675–695.
8. H. HELLER, "Relativized Polynomial Hierarchy Extending Two Levels," Ph. D. dissertation, Technischen Universität München, West Germany, 1981.
9. H. HELLER, On relativized exponential and probabilistic complexity classes, manuscript, 1983.
10. N. IMMERMAN AND S. MAHANEY, Oracles for which *NP* has polynomial size circuits, *in* "Conference on Computational Complexity Theory," Santa Barbara, March 1983, pp. 89–93.
11. R. KANNAN, A circuit size lower bound, *in* "Proc. 22nd 1981, pp. 304–309.
12. R. KARP AND R. LIPTON, Some connections between nonuniform and uniform complexity classes, *in* "Proc. 12th ACM Sympos. Theory of Comput., 1980," pp. 302–309.
13. C. LAUTEMANN, BPP and the polynomial hierarchy, *Inform. Proc. Lett.* **17** (1983), 215–217.
14. S. MAHANEY, Sparse complete sets for *NP*: Solution of a conjecture of Berman and Hartmanis, *in* "Proc. 21st IEEE Found. Comput. Sci., 1980," pp. 54–60.
15. W. PAUL, A $2.5n$ lower bound on the combinatorial complexity of Boolean functions, *SIAM J. Comput.* **6**, No. 3 (1977), 427–443.
16. W. PAUL, N. PIPPENGER, E. SZEMEREDI, AND W. TROTTER, On nondeterminism versus determinism and related problems, *in* "Proc. 24th IEEE Found. Comput. Sci., 1983," pp. 429–438.
17. C. RACKOFF, Relativized questions involving probabilistic algorithms, *J. Assoc. Comput. Mach.* **29**, No. 1 (1983), 261–268.
18. M. SIPSER, A complexity theoretic approach to randomness, *in* "Proc. 15th ACM Sympos. Theory of Comput., 1983," pp. 330–335.
19. C. WILSON, "Relativization, Reducibilities, and the Exponential Hierarchy," Univ. of Toronto TR-140, April 1980.